Qubit Manipulation in Quantum Circuits - Solving Grover's Algorithm for 2- and 3-Qubit Systems.

Purbadri Ray Chaudhuri, Kalash Sanjay Kothari

(e-mail: prc0695@gmail.com, kotharikalash05@gmail.com)

***Abstract*—Quantum computing is not just a futuristic buzzword - it's a radically new way of thinking about computation, grounded in the counterintuitive principles of quantum mechanics. In this paper, we explore the foundational building blocks of quantum computing, focusing on the concept of the qubit and the gates that manipulate its state across complex vector spaces. We begin by breaking down how qubits differ from classical bits, then delve into the mathematics and visual intuition behind key quantum gates like the Pauli, Hadamard, rotation and CNOT gates. Through detailed matrix operations and state transformations, we show how these gates form the basis of quantum circuits. The paper also analyzes Grover's Algorithm, detailing its use of superposition, phase inversion and amplitude amplification to achieve quadratic speedup in unstructured search problems. Recognizing the hardware constraints of current quantum machines, we examine circuit knitting techniques as a practical workaround for running large-scale quantum circuits on today's limited quantum hardware. By connecting rigorous mathematical concepts with real-world implementation challenges, this paper aims to provide both a deep and practical understanding of quantum computing through the lens of qubit manipulation and algorithm design.**

***Index Terms*—Quantum Computing, Qubit, Superposition, Quantum Entanglement, Qubit Manipulation, Quantum Gates, Hadamard Gate, CNOT Gate, Quantum Circuits, Quantum Algorithms, Quantum State, Measurement, Quantum Logic Gates, Quantum Decoherence, Qubit Initialization, Quantum Register, Quantum Error Correction, Reversible Computation, Quantum Bit Rotation, Quantum Information Theory.**

I. Introduction

Over the last few decades, quantum computing has emerged as a transformative paradigm that challenges the foundational limits of classical computation. At the heart of this revolution is the qubit, or quantum bit, which unlike a classical bit that can be either 0 or 1, can exist in a superposition of both states. This unique property, along with entanglement (where qubits become interconnected such that the state of one influences the state of another) and quantum interference, allows quantum systems to process information in fundamentally different and more powerful ways.

Quantum computers operate by manipulating qubits through sequences of quantum gates, which are reversible, unitary operations represented mathematically by matrices acting on complex Hilbert spaces. These gates form the building blocks of quantum circuits, much like logic gates do in classical circuits. Unlike classical algorithms, quantum algorithms evolve through superposed and entangled states, requiring precise control over qubit states and interactions.

One powerful framework within quantum algorithm design is the oracle-based query model, where an unknown function—called an oracle—is queried to extract information efficiently. In classical computing, querying such a function often requires examining inputs one by one. In contrast, quantum systems can query all possible inputs simultaneously by preparing a superposition of states, leading to dramatic improvements in search and decision problems. Algorithms like Grover's and Simon's showcase how quantum circuits and oracles combine to solve problems exponentially faster than classical methods.

However, realizing these advantages is non-trivial. It requires not only a solid theoretical framework of how qubit manipulation translates to algorithmic advantage but also a deep understanding of how such operations are implemented in physical quantum computing systems. Various hardware platforms—such as superconducting qubits, trapped ions, and photonic systems—offer different capabilities and limitations in terms of coherence time, gate fidelity, and circuit depth.

This paper investigates the intersection of these domains: the mathematical foundations of qubit manipulation, its role in oracle-based query models, and how these ideas translate into the physical realizations of modern quantum computing systems. By bridging theoretical constructs with experimental implementations, we aim to understand how fundamental quantum operations enable superior algorithmic performance and what challenges remain in building scalable, reliable quantum processors.

II. Working of a Quantum Computer

Quantum computers operate fundamentally differently from classical computers by utilizing the principles of quantum mechanics such as superposition, entanglement, and interference. At the core of a quantum computer lies the ability to manipulate quantum information in ways that allow for certain computations to be performed more efficiently than is possible classically. These computations are carried out by encoding information in quantum bits (qubits) and manipulating them using quantum gates arranged in quantum circuits.

A. Quantum Bits

A quantum bit, or qubit, is the fundamental unit of quantum information. Unlike a classical bit, which can exist only in one of two states, 0 or 1, a qubit can exist in a linear superposition

of both. Mathematically, a qubit is described as: The states $|0\rangle$ and $|1\rangle$ are usually represented as

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

According to the Dirac notation, we get

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,, \quad \text{where } \alpha, \beta \in \mathbb{C} \text{ and } |\alpha|^2 + |\beta|^2 = 1.$$

Here, $|0\rangle$ and $|1\rangle$ are the computational basis states, and α and β are complex coefficients known as probability amplitudes. When a qubit is measured, the outcome will be $|0\rangle$ with probability $|\alpha|^2$ and $|1\rangle$ with probability $|\beta|^2$. This probabilistic nature of measurement is a key departure from classical determinism.

A pure qubit state can be represented on the Bloch sphere, a geometric representation in which every point on the surface corresponds to a possible qubit state. This visualization helps in understanding how quantum gates rotate and transform qubits in a three-dimensional space, emphasizing the continuous nature of quantum states.

B. Quantum Gates

Quantum gates are the basic building blocks of quantum circuits. They perform deterministic and reversible transformations on qubits and are represented mathematically by unitary matrices. When applied to a qubit or a group of qubits, these gates evolve the state according to the rules of quantum mechanics, specifically preserving the total probability (i.e., they maintain normalization).

graphicx array booktabs amsmath

C. Mathematical and Visual Representation of the Gates Acting over the Qubit

In the earlier section, we came to know what a qubit is, how it works, and what the various gates are that help us achieve the computing we desire. That is, we just explored all the required basic building blocks of quantum computing. Now, using these building blocks, we build separate simple units, which, when put together, make a quantum system. These simple units are quantum circuits. Now, the next question is how these circuits are actually built. This has a simple answer: we use the quantum gates, which have their effect on the qubit and transform it. Now what are these transformations that happen? Mathematically, these transformations seem like linear transformations, which are done with help of the different matrices. Each gate has its own matrix, which is used a function applied to the state of the qubit. These applied functions transform the state of the qubit representing different states.

Few of the matrices of the respective gates have been listed in the previous section, in reference to which, in this section, we perform the transformation and show the actions of different gates on the qubits. Here, we show the matrix representations of the gates and the complete calculations for both $|0\rangle$ and $|1\rangle$ states.

TABLE I
COMMON QUANTUM GATES AND THEIR PROPERTIES

Gate Name	**Symbol**	**Matrix**	**Description**
Pauli-X (NOT)	X	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	Bit flip: $\|0\rangle \leftrightarrow \|1\rangle$
Pauli-Y	Y	$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$	Bit and phase flip
Pauli-Z	Z	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	Phase flip
Hadamard	H	$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$	Creates superposition of $\|0\rangle$ and $\|1\rangle$
Phase (S)	S	$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$	$\pi/2$ phase shift
T Gate	T	$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$	$\pi/4$ phase shift
CNOT	CNOT	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	Flips target if control is $\|1\rangle$
SWAP	SWAP	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	Swaps two qubits
Toffoli	CCNOT	—	Flips target if both controls are $\|1\rangle$
Fredkin	CSWAP	—	Swaps two targets if control is $\|1\rangle$
$R_x(\theta)$	R_x	$\begin{bmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}$	Rotation around X-axis
$R_y(\theta)$	R_y	$\begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}$	Rotation around Y-axis
$R_z(\theta)$	R_z	$\begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}$	Rotation around Z-axis
U3 Gate	$U3$	$\begin{bmatrix} \cos\frac{\theta}{2} & -e^{i\lambda}\sin\frac{\theta}{2} \\ e^{i\phi}\sin\frac{\theta}{2} & e^{i(\phi+\lambda)}\cos\frac{\theta}{2} \end{bmatrix}$	General single-qubit rotation

1) Pauli-X (NOT) Gate: The Pauli-X gate, or NOT gate, flips the state of a qubit. Its matrix representation is:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Now, let's apply the Pauli-X gate to both $|0\rangle$ and $|1\rangle$:
For $|0\rangle$:

$$X\,|0\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle$$

For $|1\rangle$:

$$X\,|1\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle$$

Thus, the Pauli-X gate flips $|0\rangle$ to $|1\rangle$ and vice versa.

2) Pauli-Y Gate: The Pauli-Y gate applies a π rotation around the Y-axis of the Bloch sphere. Its matrix representation is:

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$$

For $|0\rangle$:

$$Y|0\rangle = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ i \end{bmatrix}$$

For $|1\rangle$:

$$Y|1\rangle = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} -i \\ 0 \end{bmatrix}$$

Thus, the Pauli-Y gate introduces a phase of i when applied to $|0\rangle$ and a phase of $-i$ when applied to $|1\rangle$.

3) Pauli-Z Gate: The Pauli-Z gate applies a phase flip to the state $|1\rangle$, leaving $|0\rangle$ unchanged. Its matrix representation is:

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

For $|0\rangle$:

$$Z|0\rangle = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle$$

For $|1\rangle$:

$$Z|1\rangle = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -1 \end{bmatrix} = -|1\rangle$$

Thus, the Pauli-Z gate leaves $|0\rangle$ unchanged and flips the phase of $|1\rangle$ by -1.

4) Rotation Gate $R_x(\theta)$: The $R_x(\theta)$ gate performs a rotation around the X-axis by an angle θ. Its matrix representation is:

$$R_x(\theta) = \begin{bmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}$$

For $|0\rangle$:

$$R_x(\theta)|0\rangle = \begin{bmatrix} \cos\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} \end{bmatrix}$$

For $|1\rangle$:

$$R_x(\theta)|1\rangle = \begin{bmatrix} -i\sin\frac{\theta}{2} \\ \cos\frac{\theta}{2} \end{bmatrix}$$

Thus, the $R_x(\theta)$ gate rotates the qubit around the X-axis by the angle θ.

5) Rotation Gate $R_y(\theta)$: The $R_y(\theta)$ gate performs a rotation around the Y-axis by an angle θ. Its matrix representation is:

$$R_y(\theta) = \begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}$$

For $|0\rangle$:

$$R_y(\theta)|0\rangle = \begin{bmatrix} \cos\frac{\theta}{2} \\ \sin\frac{\theta}{2} \end{bmatrix}$$

For $|1\rangle$:

$$R_y(\theta)|1\rangle = \begin{bmatrix} -\sin\frac{\theta}{2} \\ \cos\frac{\theta}{2} \end{bmatrix}$$

Thus, the $R_y(\theta)$ gate rotates the qubit around the Y-axis by the angle θ.

6) Rotation Gate $R_z(\theta)$: The $R_z(\theta)$ gate performs a rotation around the Z-axis by an angle θ. Its matrix representation is:

$$R_z(\theta) = \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}$$

For $|0\rangle$:

$$R_z(\theta)|0\rangle = e^{-i\theta/2} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = e^{-i\theta/2}|0\rangle$$

For $|1\rangle$:

$$R_z(\theta)|1\rangle = e^{i\theta/2} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = e^{i\theta/2}|1\rangle$$

Thus, the $R_z(\theta)$ gate applies a phase shift depending on the angle θ.

7) Hadamard Gate: The Hadamard gate creates an equal superposition of $|0\rangle$ and $|1\rangle$. Its matrix representation is:

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

For $|0\rangle$:

$$H|0\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

For $|1\rangle$:

$$H|1\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

Thus, the Hadamard gate transforms a computational basis state into an equal superposition.

8) CNOT Gate: The controlled-NOT (CNOT) gate flips the target qubit if the control qubit is in state $|1\rangle$. Its matrix representation is:

$$\text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

For $|10\rangle$:

$$\text{CNOT}\,|10\rangle = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\begin{bmatrix}1\\0\\0\\0\end{bmatrix} = \begin{bmatrix}1\\0\\0\\1\end{bmatrix} = |11\rangle$$

Thus, the CNOT gate flips the target qubit when the control qubit is $|1\rangle$.

9) Phase Gate: The Phase gate adds a phase of $\frac{\pi}{2}$ to the state $|1\rangle$. Its matrix representation is:

$$S = \begin{bmatrix}1&0\\0&i\end{bmatrix}$$

For $|1\rangle$:

$$S\,|1\rangle = i\,|1\rangle$$

Thus, the Phase gate introduces a phase of i to the state $|1\rangle$.

10) T Gate: The T gate adds a phase of $\frac{\pi}{4}$ to the state $|1\rangle$. Its matrix representation is:

$$T = \begin{bmatrix}1&0\\0&e^{i\pi/4}\end{bmatrix}$$

For $|1\rangle$:

$$T\,|1\rangle = e^{i\pi/4}\,|1\rangle$$

Thus, the T gate introduces a phase of $e^{i\pi/4}$ to the state $|1\rangle$.

III. Grover's Algorithm

Grover's Algorithm, introduced by Lov Grover in 1996, represents a fundamental milestone in the development of quantum algorithms, offering a quadratic speedup for solving the unstructured search problem. The algorithm addresses the task of finding a specific input x_0 that satisfies a boolean function $f(x) = 1$, where $f : \{0,1\}^n \to \{0,1\}$, given black-box access to f. Classically, the only way to guarantee finding x_0 in an unsorted database of $N = 2^n$ entries is through exhaustive search, which requires $O(N)$ queries in the worst case. Grover's algorithm, leveraging quantum mechanical principles such as superposition, interference, and entanglement, reduces this query complexity to $O(\sqrt{N})$, thereby providing a provable quantum advantage for a wide class of problems where the structure of the search space is unknown or not exploitable.

The algorithm functions through iterative applications of two main operators: the oracle and the diffusion (or inversion-about-the-mean) operator. Initially, the quantum register is prepared in an equal superposition of all possible states using Hadamard gates, effectively distributing the probability amplitude uniformly across the entire solution space. The oracle, typically implemented as a phase-flip operator, identifies the correct solution by inverting the sign of its amplitude without altering the others. This operation does not reveal the solution directly but encodes its presence via a subtle phase change. The diffusion operator then performs an inversion about the mean of the amplitudes, amplifying the amplitude of the marked (solution) state while suppressing the others through quantum interference. Repeating this pair of operations approximately $\lfloor \frac{\pi}{4}\sqrt{N} \rfloor$ times maximizes the probability of measuring the correct answer upon observation.

What distinguishes Grover's algorithm is not merely its speedup, but its generality. It is optimal in the quantum query model for unstructured search, as proven by the BBBV theorem, and has found theoretical applications ranging from cryptanalysis (e.g., attacking symmetric-key cryptographic systems) to solving NP-complete problems more efficiently in certain bounded cases. Moreover, its conceptual design—centered around amplitude amplification—has led to the development of a broader class of quantum algorithms that generalize or adapt its techniques for other optimization and decision problems.

This paper provides a comprehensive examination of Grover's algorithm, beginning with its mathematical formulation and circuit-level implementation.

IV. Mathematical Foundations of Grover's Algorithm

Grover's algorithm operates within the framework of quantum mechanics and linear algebra. It is designed to find a unique marked item from an unsorted database of $N = 2^n$ items using only $O(\sqrt{N})$ queries. The mathematical foundation of this algorithm is built on quantum state superposition, unitary transformations, and amplitude amplification.

A. State Space and Initialization

Consider an n-qubit system. The Hilbert space of this system is $\mathbb{C}^{2^n}$, spanned by 2^n computational basis states $\{|0\rangle, |1\rangle, \ldots, |2^n - 1\rangle\}$.

Initially, the quantum register is prepared in the state $|0\rangle^{\otimes n}$. To create a uniform superposition over all possible basis states, we apply the Hadamard gate $H^{\otimes n}$ to each qubit:

$$|\psi_0\rangle = H^{\otimes n}\,|0\rangle^{\otimes n} = \frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle \tag{1}$$

This state gives each possible solution an equal probability amplitude.

B. Oracle Operator (Phase Inversion)

Grover's algorithm assumes the existence of a quantum oracle O_f that recognizes the correct solution. The oracle is defined as a unitary operator that flips the phase of the marked state $|x^*\rangle$:

$$O_f\,|x\rangle = \begin{cases} -\,|x\rangle & \text{if } x = x^* \\ |x\rangle & \text{otherwise} \end{cases} \tag{2}$$

This is often referred to as a phase query or phase inversion gate, and it forms the first step of the Grover iteration.

C. Diffusion Operator (Inversion About the Mean)

After the oracle has inverted the amplitude of the correct state, we apply the diffusion operator D to amplify the probability amplitude of the marked state. It is defined as:

$$D = 2\left|\psi_0\right\rangle\left\langle\psi_0\right| - I \tag{3}$$

This operator inverts the amplitude of each state about the average amplitude of all states. In practice, it can be implemented using Hadamard gates, Pauli-X gates, and a multi-controlled Z gate.

D. Grover Iterations

A single Grover iteration G consists of applying the oracle followed by the diffusion operator:

$$G = DO_f \tag{4}$$

After approximately $k = \left\lfloor \frac{\pi}{4}\sqrt{N} \right\rfloor$ iterations, the probability of measuring the correct solution becomes very high.

E. Geometric Interpretation

The algorithm's operation can be understood geometrically as a rotation in a 2D subspace spanned by the marked state $|x^*\rangle$ and the superposition of the unmarked states. Each Grover iteration rotates the state vector closer to $|x^*\rangle$.

F. Optimality: The BBBV Theorem

According to the Bennett–Bernstein–Brassard–Vazirani (BBBV) theorem, no quantum algorithm can solve the unstructured search problem using fewer than $\Omega(\sqrt{N})$ queries, proving that Grover's algorithm is asymptotically optimal in this setting.

V. Application of Circuit Knitting Techniques to Grover's Algorithm

Quantum computing is currently limited by hardware constraints such as a small number of qubits, high error rates, and restricted qubit connectivity. These limitations make it challenging to run large and complex quantum circuits on near-term quantum devices. To address this, researchers have developed circuit cutting and circuit knitting techniques, which allow large quantum circuits to be divided into smaller, more manageable subcircuits. These smaller subcircuits can be executed independently on current quantum hardware and their results combined using classical computation to estimate the outcome of the original, unpartitioned circuit. This process significantly extends the practical computational reach of quantum devices with limited resources.

Circuit cutting involves intentionally breaking a quantum circuit at specific points. These break points can occur either along qubit wires (wire cuts) or at multi-qubit gates (gate cuts). A wire cut interrupts the temporal continuity of a qubit's state, while a gate cut separates qubits involved in the same quantum gate. Both types of cuts introduce dependency violations that must be resolved to maintain the validity of the computation. This is where circuit knitting techniques come into play. Circuit knitting provides a systematic way to handle these violations, allowing the subcircuits to be executed independently while preserving the overall logic of the original quantum computation.

Gate cuts are typically resolved using quasiprobability decomposition (QPD), which represents a quantum gate as a weighted sum of simpler, local operations. Each subcircuit execution randomly samples from these local operations according to their weights, and the results are classically reweighted to reconstruct the original computation. Wire cuts can be handled by transferring the quantum state to another qubit, often using quantum teleportation, swap gates, or optimized "move" circuits. These operations may require ancillary qubits and classical communication between subcircuits. The efficiency of circuit knitting largely depends on the number and location of the cuts, as well as the availability of ancilla qubits and classical communication. Sampling overhead, which reflects the number of times subcircuits must be executed to achieve a reliable estimate, increases exponentially with the number of cuts. Therefore, advanced optimization methods are employed to minimize the overhead by selecting cut points with minimal computational cost.

Grover's algorithm, which is designed for searching an unsorted database in $\mathbf{O}(\sqrt{N})$ time, consists of repeated applications of an oracle and a diffusion operator. These components involve highly entangled multi-qubit operations, making the circuit deep and qubit-intensive as the problem size grows. Applying circuit knitting to Grover's algorithm is a promising approach for scaling the algorithm on limited hardware. For example, the oracle, which marks the target solution by inverting its phase, often uses controlled operations that span many qubits and can be partitioned through gate cuts. Similarly, the diffusion operator, which amplifies the amplitude of the target state, can be split using wire cuts that relocate state information across partitions. By carefully applying circuit knitting, individual pieces of Grover's algorithm can be tested and run on hardware that does not have enough qubits to handle the full circuit at once.

The main advantage of using circuit knitting for Grover's algorithm lies in its potential to reduce the hardware requirements without compromising the ability to simulate larger problem instances. This enables researchers to validate and study scaled-up versions of Grover's search even with the constraints of near-term quantum devices. Furthermore, smaller subcircuits tend to be less error-prone, and the classical postprocessing step involved in circuit knitting can efficiently combine their results. While the sampling overhead remains a concern, optimized partitioning strategies such as those presented in the referenced work significantly reduce this burden by combining gate and wire cuts, using ancilla qubits, and employing classical communication. The ability to reconstruct Grover's search through partitioned execution not only allows for more extensive experimentation with quantum search but also provides a pathway to scaling quantum algorithms beyond the current physical limitations of quantum computers.

VI. Conclusion

Quantum computing marks a paradigm shift in computational science by utilizing the principles of superposition and entanglement to surpass classical limits. This work has detailed the essential mechanisms of qubit control and the implementation of quantum gates to build efficient quantum circuits. It has also highlighted the significance of advanced algorithms such as Grover's in demonstrating the power of quantum speedup. Although current quantum hardware faces challenges in scalability, coherence, and error rates, emerging techniques like circuit knitting offer practical strategies to mitigate these limitations. By partitioning complex quantum circuits into smaller, executable subcircuits, circuit knitting allows researchers to simulate and test large-scale algorithms using today's constrained devices. Continued progress in both theoretical understanding and experimental realization—particularly in techniques that integrate quantum and classical computation—will be crucial to fully harness the transformative capabilities of quantum computing across a broad range of scientific and technological applications.

References

[1] L. K. Grover, *A fast quantum mechanical algorithm for database search*, Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC), 1996, pp. 212–219.

[2] P. Benioff *The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines*, Journal of Statistical Physics, 22, 1980, pp. 563-591.

[3] D.Deutsch *Quantum Theory, the Church-Turing Principle, and the Universal Quantum Computer*, Proc. Royal Society London Ser. A, 400, 1985, pp. 96-117.

[4] Sebastian Brandhofer, Ilia Polian, Kevin Krsulich *Optimal Partitioning of Quantum Circuits using Gate Cuts and Wire Cuts* arXiv:2308.09567, https://arxiv.org/abs/2308.09567